Inapproximability Results for Guarding Polygons without Holes*

Stephan Eidenbenz

Institute for Theoretical Computer Science, ETH 8092 Zürich, Switzerland
eidenben@inf.ethz.ch

Abstract. The three art gallery problems Vertex Guard, Edge Guard and Point Guard are known to be NP-hard [8]. Approximation algorithms for Vertex Guard and Edge Guard with a logarithmic ratio were proposed in [7]. We prove that for each of these problems, there exists a constant $\epsilon > 0$, such that no polynomial time algorithm can guarantee an approximation ratio of $1 + \epsilon$ unless $P = NP$. We obtain our results by proposing gap-preserving reductions, based on reductions from [8]. Our results are the first inapproximability results for these problems.

1 Introduction and Problem Definition

Guarding polygons is a variant of the art gallery problem, which asks how many guards are needed to see every point in the interior of a polygon P given as a linked list of n points in the $x-y$-plane. Polygon guarding problems are classified as to where the guards may be positioned, what kind of guards can be used, whether only the boundary or all of the interior of the polygon should be seen from at least one guard, and assumptions are also made on certain properties of the input polygon. In this paper, we assume that the input polygons are simple, i.e., such that no two nonconsecutive edges intersect. A point *sees* some other point, if the line segment connecting the two points does not intersect the exterior of the polygon P.

Many results are known concerning upper and lower bounds on the number of guards needed. Comparatively few papers study the computational complexity of art gallery problems. Surveys on the general topic of art galleries include [11,13], and [14]. [9] contains an overview of what is known about the computational complexity of several art gallery problems. As for the computational complexity of art gallery problems, it is known that the problem of covering a polygon with holes with a minimum number of convex polygons or star-shaped polygons is NP-hard [10]. The latter problem is equivalent to the Point Guard problem to be defined later. These problems remain NP-hard even if the input polygon has no holes (for convex polygons [4] and for star-shaped polygons [8]). The two problems Vertex Guard and Edge Guard (to be defined later) are NP-hard for polygons without holes [8]. Point Guard, Vertex Guard and Edge Guard

* This work is partially supported by the Swiss National Science Foundation

K.-Y. Chwa and O. H. Ibarra (Eds.): ISAAC'98, LNCS 1533, pp. 427–437, 1998.

for polygons with holes cannot be approximated by any polynomial algorithm within a factor $\frac{1-\epsilon}{28} \ln n$ for any $\epsilon > 0$ unless $NP \subseteq TIME(n^{O(\log \log n)})$ [6]. Finally, approximation algorithms for Vertex Guard and Edge Guard, which achieve approximation ratios of $O(\log n)$, exist [7]. The approximation algorithms work for polygons with and without holes.

We study the following problems:

- Vertex Guard(VG):
  Given a polygon P (without holes) with n vertices, find a minimum subset S of vertices such that every point on the boundary of the polygon P can be seen from at least one vertex in S. We say that guards are placed at the vertices in S. The vertices in S are called vertex guards.
- Edge Guard (EG):
  Given a polygon P (without holes) with n vertices, find a minimum subset S of edges, i.e. line segments of the polygon, such that every point on the boundary of the polygon P can be seen from at least one point on an edge in S. The edges in S are called edge-guards.
- Point Guard (PG):
  Given a polygon P (without holes) with n vertices, find a minimum subset S of points in the interior of the polygon such that every point on the boundary of the polygon P can be seen from at least one point in S. The points in S are called point-guards.

Note that our definitions differ from the corresponding definitions in [8] because in [8] the guards must see all of the interior of the polygon rather than only the boundary. It will be easy to see that our results carry over to these problems.

Since these problems are NP-hard, we would like to know how well they can be approximated by polynomial time algorithms. Urrutia points out that such results are needed [14]. When trying to determine the approximation properties of a problem, we always have two options; one of them is to find approximation algorithms that achieve a certain approximation ratio, as has been done for VG and EG [7]. The other option is to find lower bounds on the approximation ratio achievable, which is what we pursue in this paper. The result of this paper is that VG, EG and PG for polygons without holes are APX-hard, which means that for each of these problems, there is a constant $\epsilon > 0$ such that an approximation ratio of $1 + \epsilon$ cannot be guaranteed by any polynomial time algorithm, unless $NP = P$. (See [3] for an introduction to the class APX.)

We prove the results by describing a reduction from 5-Occurrence-3-Sat, which is the version of 3-SAT with each clause containing at most three literals and with each variable appearing in at most five literals. 5-Occurrence-3-Sat is $MAXSNP$-complete [12], which means that there is a constant $\gamma > 0$ such that no polynomial time algorithm can guarantee an approximation-ratio of $1+\gamma$ for 5-Occurrence-3-Sat, unless $NP = P$ [2]. Our reduction follows the lines of the reductions in [8]. We show that our reduction is gap-preserving, using a technique introduced in [2].

In Sect. 2, we propose a construction that takes a 5-Occurrence-3-Sat instance as input and yields a polygon without holes, which is a PG-instance.

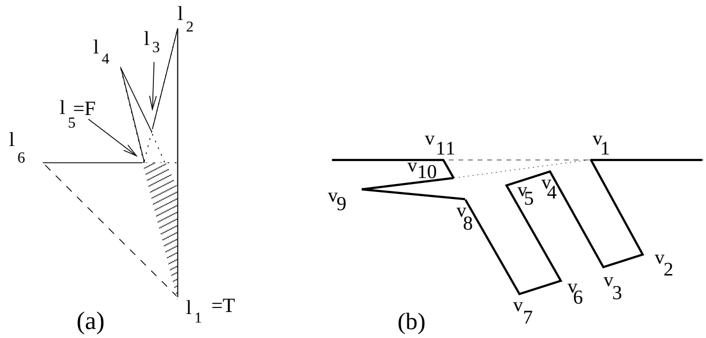

Fig. 1. (a) Literal Pattern; (b) Variable Pattern

The transformation of the solution is described in Sect. 3. We analyze the reduction and obtain our main result in Sect. 4. Section 5 contains some concluding remarks.

2 Construction of the Reduction

We present the construction for PG. Suppose we are given an instance I of 5-Occurrence-3-Sat. Let I consist of n variables $x_1, \cdots, x_n$ and of $m \leq \frac{5}{3}n$ clauses $c_1, \cdots, c_m$. We construct a polygon P that can be used in the reduction.

For every literal y_j, which is either x_j or $\neg x_j$, we construct a "literal pattern" as shown in Fig. 1(a). The literal pattern is the polygon defined by the points $l_1, \cdots, l_6$. The edge from l_6 to l_1 is not part of the final polygon, but serves as an interface to the outside of the literal pattern. All other edges in the literal pattern are part of the final polygon. The points l_4, l_5, l_1 lie on a straight line. Therefore, a guard at point l_1 or point l_5 sees all of the interior of the literal pattern. The final construction is such that a guard at point l_1 implies that the literal is true and such that a guard at point l_5 implies that the literal is false. We, therefore, call point l_1 simply T; similarly, l_5 is called F. For a finite number of guards, in order to completely see the whole literal pattern of Fig. 1(a), at least one guard must be inside the literal pattern. (If point l_4 of the pattern can be seen from a guard outside the pattern, which is possible by a guard on the line through l_4 and l_1, then in order to see the points on the segment from l_4 to l_3 from outside the pattern, an infinite number of guards is needed.)

For every clause c_i consisting of the literals y_j, y_k and y_l, we construct a "clause junction" as shown in Fig. 2. The clause junction is the polygon starting at point p_1 and moving along the solid line through p_2, p_3, p_4, the three literal patterns and ending at point p_8. The line from p_8 to p_1 is not part of the final polygon, but, again, serves as an interface of the clause junction to the outside. At least three guards are needed to completely see the clause junction in Fig. 2: one guard for each literal pattern. At least one of these three guards needs to be

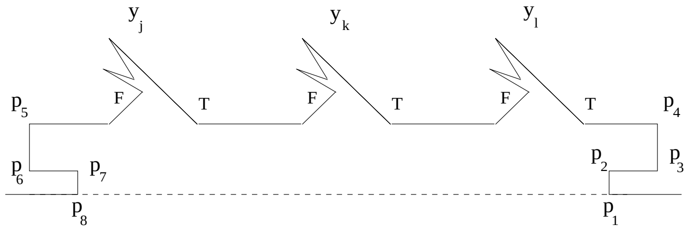

Fig. 2. Clause Junction

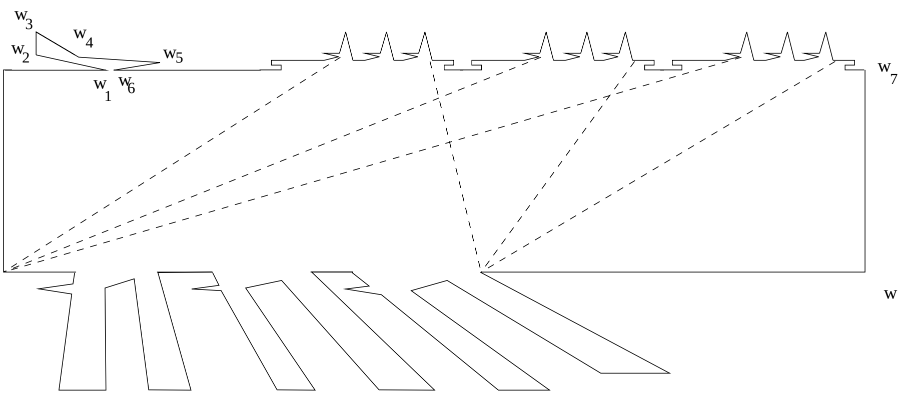

Fig. 3. Putting the pieces together

at point $l_1(= T)$ of the corresponding literal pattern in order to see point p_6 of the clause junction.

For every variable x_j, we construct a "variable pattern" as shown in Fig. 1(b). The variable pattern is the polygon defined by points $v_1, \cdots, v_{11}$. The edge from v_1 to v_{11} is not part of the final polygon, but serves as an interface. We call the polygon defined by the points v_1, v_2, v_3, v_4 the *right leg* of the variable pattern and the polygon defined by v_5, v_6, v_7, v_{11} the *left leg* of the variable pattern. The polygon formed by v_8, v_9, v_{10} is called the *triangle* of the variable pattern. The points v_9, v_{10}, v_1 lie on a straight line. Therefore, for a finite number of guards, in order to completely see the triangle (formed by the three points v_8, v_9, v_{10}), at least one guard must be inside the variable pattern. In the final polygon, this guard sits at point v_1, if the variable is assigned the value true, and it sits at point v_5, if the variable is false.

We put all pieces together as shown in Fig. 3. A guard at point w_1 sees all the legs of the variable patterns. The points w_3, w_4, w_1 are in a straight line. For a finite number of guards, in order to completely see the triangle w_1, w_2, w_3, at least one guard must lie inside the ear-like feature defined by the polygon $w_1, \cdots, w_6$. Finally, we construct for each literal y_j in each clause two "spikes" as shown in Fig. 4. Figure 4 (a) is for the case, when literal y_j is equal to x_j; Fig-

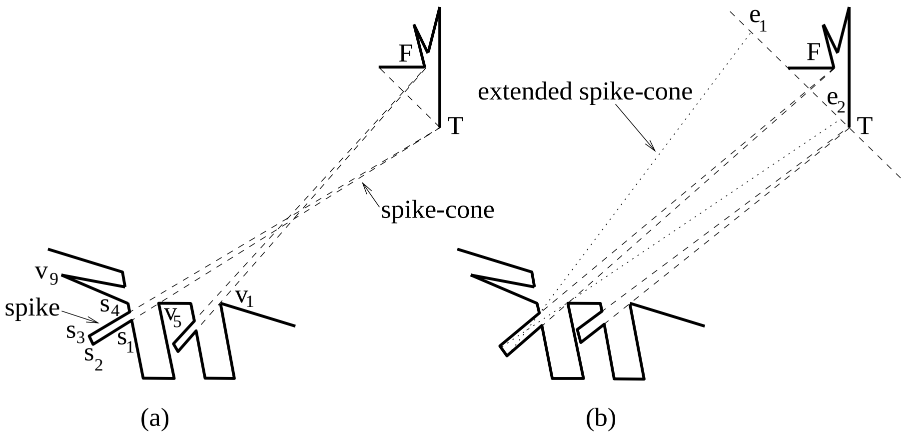

Fig. 4. (a): Spikes, when literal y_j is equal to x_j; (b): Spikes, when literal y_j is equal to $\neg x_j$

ure 4 (b) is for the case, when literal y_j is equal to $\neg x_j$. The points s_2, s_1 and $l_5(= F)$ of the corresponding literal pattern are in a straight line, as are s_3, s_4 and l_5. A *spike* is a polygon defined by points s_1, s_2, s_3, s_4. The corresponding *spike-cone* is defined as the triangle s_1, s_4, l_1 or s_1, s_4, l_5. (Note that $l_1 = T$ and $l_5 = F$.) The points s_2, s_4, e_1 are in a straight line as are points s_3, v_1, e_2. We call the polygon s_3, v_1, e_2, e_1 the *extended spike-cone* of the corresponding spike (see Fig. 4 (b)). For a guard outside the variable pattern, in order to see point s_2 of a spike, it is a necessary condition that this guard lie in the corresponding extended spike-cone. We, therefore, call point s_2 of each spike the *distinguished point* of the spike. Note that the spikes may have to be very thin (thinner than indicated in Fig. 4), since up to five spikes must fit into each leg without intersecting each other.

In order to ensure that at no point between the variable and the literal patterns a guard sees the distinguished points of three or more spikes that belong to three different legs of variable patterns, we construct the polygon in such a way that no three extended spike-cones of different legs of variable patterns intersect. This restriction forces us to give a more detailed description of the whole construction.

First, fix the points $l_1(= T)$ and $l_5(= F)$ of each literal pattern on a horizontal line with constant distance between them. (We will move the F points above the line at the end of the construction.) For each l_1 and l_5 we fix the point e_1 at distance a to the left of the point l_1 or l_5. We also fix point e_2 at distance $2a$ to the right of the point l_1 or l_5 with a a constant as indicated in Fig. 5 (a). The constant a has to be small enough, in order to avoid that the extended spike-cones of two neighboring points l_1 and l_5 intersect at the horizontal line of the literal patterns, and in order to avoid that spikes in the same leg of a variable pattern intersect. Choose w_1 at a constant distance to the left of the

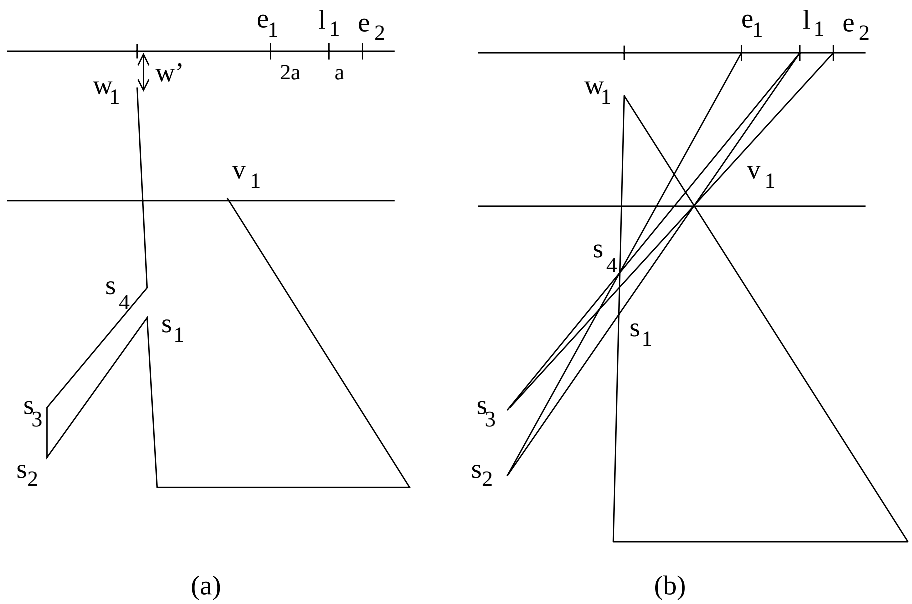

Fig. 5. Construction of the right leg

leftmost l_5 (of the leftmost literal) and at a constant distance w' below the line of the literal patterns.

Assume that the variable patterns for the variables $x_1, \cdots, x_{i-1}$ have already been constructed. We show how to construct the next variable pattern for variable x_i. We determine point v_1 of the variable pattern as follows: We determine the rightmost point l_5 or l_1 of the at most five literal patterns that are literals of variable x_i, from which a spike must be constructed in the right leg of the variable pattern. W. l. o. g. assume that this point is some point l_1. We draw a line from point e_2 (with respect to l_1) through the point S, which is the point with largest y-value of all points where two extended spike-cones intersect. Let v_1' be the point, where this line and the horizontal line of the variable patterns intersect. Now, let v_1 be at some constant distance to the left of v_1'. We now construct the right leg of the variable pattern as indicated in Fig. 5 (a), which shows the right leg with the top-most spike. Figure 5 (b) shows how (a) is constructed. We describe this step by step. Once point v_1 has been determined, draw a line segment from point e_2 through v_1 and stop at a certain, fixed x-distance from v_1. This yields point s_3. Draw a line segment from l_1 through v_1 and stop at the same x-distance from v_1, which is point s_2. Draw a line segment from s_3 to l_1. Then, draw a line segment from s_2 to e_1. Point s_4 is the intersection point of these two line segments. Finally, draw the leg of the variable pattern by drawing a line segment from w_1 through v_1 and a line segment from w_1 through s_4, which yields point s_1. We continue with the remaining spikes of the leg. The remaining spikes are drawn with x-distances from points s_4 to points s_2 always the same.

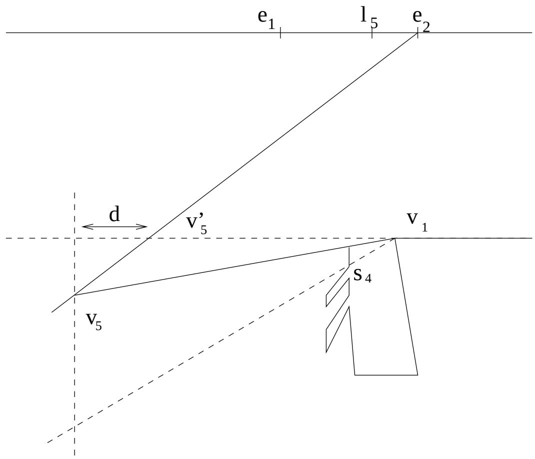

Fig. 6. The left leg

We construct the left leg of the variable pattern accordingly, except for point v_5. It is shown in Fig. 6 how to obtain v_5; v_5 is the intersection point of a vertical line at a constant distance d from v_5' with either the extended spike-cone borderline from e_2 through v_5' or with the line from v_1 through point s_4 (of the top-most spike) of the first leg. v_5' is obtained using the same procedure as for point v_1.

Once we have constructed all variable patterns, we need to construct the literal patterns as shown in Fig. 7. In a similar method as used for the left leg of the variable pattern, we move point l_5 upwards. d' is a constant. This operation can be performed in such a manner that the extended spike-cone remains unchanged (or only "shrinks" in size) and that the spike-cone only changes in a way, which results in a smaller opening of the spike in the variable pattern. Finally, we construct the clause junctions as shown in Fig. 8. The clause junctions are constructed such that no spike-cone intersects with the polygon boundary. Distance b is defined to be the minimum of the y-distance of the point with maximum y-value among all points, where two extended spike-cones intersect, from the horizontal line of the literal patterns, and the distance w' (see Fig. 5). We complete the construction as indicated in Fig. 3; points w_1, w_6 and p_8 of the leftmost clause junction are in a straight line.

An analysis reveals that the coordinates of all points can be computed in polynomial time; some coordinates require a polynomial number of bits. (A similar analysis can be found in full detail in [5].) Therefore, the construction is polynomial in the size of the input.

3 Transformation of the Solution

We describe how to obtain an assignment of the variables of the satisfiability instance, given a feasible solution of the corresponding PG instance.

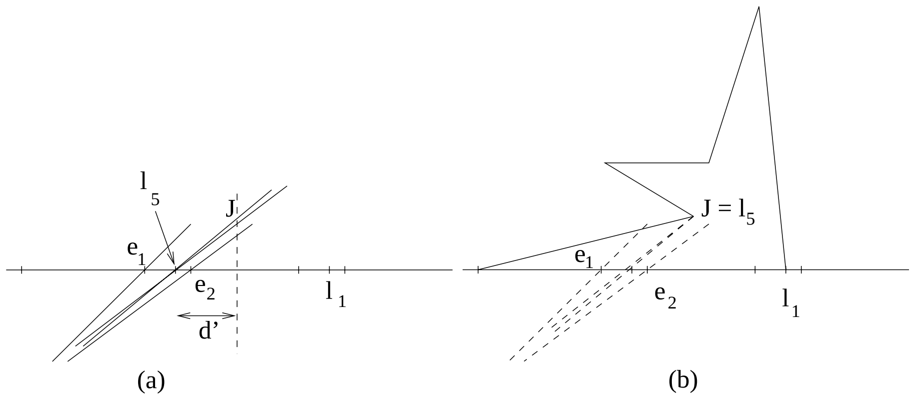

Fig. 7. Construction of the literal patterns

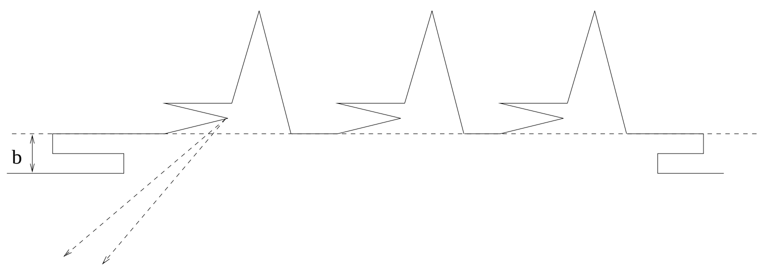

Fig. 8. Construction of the clause junctions

– Determine which guard is inside the polygon $w_1, \cdots, w_6$ (see Fig. 3) and move this guard to w_1.
– For each literal pattern, determine which guard sees point l_4 (and is inside the literal pattern) and move it to the closer one of the two points $l_1(=T)$ or $l_5(=F)$ of the literal pattern. If there is a guard at both points $l_1(=T)$ and $l_5(=F)$ of the literal pattern, move the guard at $l_5(=F)$ along the spike-cone to the point v_1 or v_5 of the corresponding variable pattern. If there is more than one guard at point $l_1(=T)$ or $l_5(=F)$, move all but one guard along the spike-cone to the variable pattern.
– For each clause junction, move any guard that sees point p_6 of the junction to the closest point T of any literal pattern in the clause junction.
– For each variable pattern, move the guard that sees point v_9 of the variable pattern to the closest of the two points v_1 and v_5 of the variable pattern. If there are additional guards in the area of the variable pattern, move them to v_1 or v_5 depending upon which leg of the variable pattern they are in.
– Move all guards that lie in an extended spike-cone, but not at point l_1 or l_5 of a literal pattern to point v_1 or v_5 of the corresponding variable pattern. If a guard lies in an intersection of two extended spike-cones that belong

to different legs (of variable patterns), add a guard and move these guards to the corresponding points v_1 or v_5 in the variable patterns. Intersections of three extended spike-cones that belong to three different legs of variable patterns do not exist.
- Guards at other points can be moved to any point v_1 or v_5 of any variable pattern, if there is no guard at v_1 or v_5 already. Guards in intersections of two spike-cones of the same variable pattern are moved to point v_1 or v_5 of the variable pattern, if there is no guard there already.

The solution obtained by moving and adding guards as indicated is still feasible, i.e., the guards still see all of the polygon.

We are now ready to set the truth values of the variables. For each variable pattern x_j, if there is a guard at point v_5 and no guard at point v_1, let x_j be false. If there is a guard at point v_1 and no guard at point v_5, let x_j be true. If there is a guard at both v_1 and v_5, then set x_j in such a way that a majority of the literals of x_j become true.

4 Analysis of the Reduction

Consider the promise problem of 5-Occurrence-3-Sat, where we are given an instance of 5-Occurrence-3-Sat, and we are promised that the instance is either satisfiable or at most $m(1-4\epsilon)$ clauses are satisfiable by any assignment of the variables. The NP-hardness of this problem for small enough values of ϵ follows from the fact that 5-Occurrence-3-Sat is $MAXSNP$-complete (see [12] and [2]). An analysis of our reduction leads to the following.

Theorem 1. *Let I be an instance of the promise problem of* 5-Occurrence-3-Sat*, let n be the number of variables in I and let $m \leq \frac{5}{3}n$ be the number of clauses in I. Let $OPT(I)$ denote the maximum number of satisfiable clauses (for any assignment). Furthermore, let I' be the corresponding instance of PG and let $OPT(I')$ denote the minimum number of guards needed to completely see the polygon of I'. Then, the following hold:*

- *If $OPT(I) = m$, then $OPT(I') \leq 3m + n + 1$.*
- *If $OPT(I) \leq m(1-4\epsilon)$, then $OPT(I') \geq 3m + n + 1 + \epsilon m$.*

Theorem 1 shows that our reduction is gap-preserving (see [2]). It shows that the promise problem of PG with parameters $3m+n+1$ and $3m+n+1+\epsilon m$ is NP-hard. Note that $m \geq \frac{n}{3}$, since each variable appears as a literal at least once. Therefore, unless $NP = P$, no polynomial time approximation algorithm for PG can achieve an approximation ratio of:

$$\frac{3m+n+1+\epsilon m}{3m+n+1} = 1 + \frac{\epsilon}{3 + \frac{n+1}{m}} \geq 1 + \frac{\epsilon}{3 + \frac{3(n+1)}{n}} \geq 1 + \frac{\epsilon}{7}$$

Thus, we have our main result:

Theorem 2. *The* Point Guard *problem for polygons without holes is APX-hard.*

Note that our proof works as well for the PG problem as defined in [8], where it is required that the guards see all of the interior and the boundary of the polygon.

5 Discussion

With similar arguments, we can show that the Vertex Guard and Edge Guard problems for polygon without holes are APX-hard as well. This characterization of the approximability of Vertex Guard, Point Guard and Edge Guard for polygons without holes is not the end of the story. We know of no approximation algorithm that achieves a constant ratio. We, therefore, did not focus on giving a concrete value for the inapproximability ratio. As long as no constant ratio approximation algorithms are known, it suffices to show that these problems are APX-hard. The approximation algorithms in [7] only achieve ratios of $O(\log n)$ for VG and EG.

References

1. P. Bose, T. Shermer, G. Toussaint and B. Zhu; *Guarding Polyhedral Terrains*; Computational Geometry 7, pp. 173-185, Elsevier Science B. V., 1997.
2. S. Arora, C. Lund; *Hardness of Approximations*; in: Approximation Algorithms for NP-Hard Problems (ed. Dorit Hochbaum), PWS Publishing Company, pp. 399-446, 1996. 428, 435
3. D. P. Bovet and P. Crescenzi; *Introduction to the Theory of Complexity*; Prentice Hall, 1993. 428
4. J. C. Culberson and R. A. Reckhow; *Covering Polygons is Hard*; Proc. 29th Symposium on Foundations of Computer Science, 1988. 427
5. St. Eidenbenz, Ch. Stamm, P. Widmayer; *Positioning Guards at Fixed Height above a Terrain - an Optimum Inapproximability Result*; to appear in Proceedings ESA, 1998. 433
6. St. Eidenbenz, Ch. Stamm, P. Widmayer; *Inapproximability of some Art Gallery Problems*; to appear in Proceedings CCCG, 1998. 428
7. S. Ghosh; *Approximation Algorithms for Art Gallery Problems*; Proc. of the Canadian Information Processing Society Congress, 1987. 427, 428, 436
8. D. T. Lee and A. K. Lin; *Computational Complexity of Art Gallery Problems*; in IEEE Trans. Info. Th, pp. 276-282, IT-32 (1986). 427, 428, 436
9. B. Nilsson; *Guarding Art Galleries - Methods for Mobile Guards*; doctoral thesis, Department of Computer Science, Lund University, 1994. 427
10. J. O'Rourke and K. J. Supowit; *Some NP-hard Polygon Decomposition Problems*; IEEE Transactions on Information Theory, Vol IT-29, No. 2, 1983. 427
11. J. O'Rourke; *Art Gallery Theorems and Algorithms*; Oxford University Press, New York (1987). 427
12. C.H. Papadimitriou and M. Yannakakis; *Optimization, Approximation, and Complexity Classes*; Proc. 20th ACM Symposium on the Theory of Computing, 1988. 428, 435

13. T. Shermer; *Recent Results in Art Galleries*; Proc. of the IEEE, 1992. 427
14. J. Urrutia; *Art Gallery and Illumination Problems*; to appear in Handbook on Computational Geometry, edited by J.-R. Sack and J. Urrutia, 1998. 427, 428